**Question 1 (20 marks)**  Batched range reporting is the following problem: Given a set $R$ of rectangles in the plane and a set $P$ of points in the plane, report all pairs $(p, r)$ such that $p \in P$, $r \in R$, and $p \in r$. If $N = |P| + |R|$ and $K$ is the number of such pairs $(p, r)$ reported, your task is to design an algorithm that solves this problem using $O(\text{sort}(N) + K/B)\ I/Os$. Provide a description, analysis, and proof of correctness of your algorithm.

**Question 2 (50 marks)**

(a) Orthogonal range reporting is the problem of storing a set $P$ of points in a data structure so that, for any query rectangle $r$, all points in $P$ that lie inside $r$ can be reported efficiently. In class, we showed that the $k$-d-tree uses linear space and achieves a query bound of $O(\sqrt{N}/B + K/B)\ I/Os$ for points in the plane, where $N = |P|$ and $K$ is the number of reported points. Describe briefly the modifications to the structure that are necessary to achieve a query bound of $O((N/B)^{1-1/k} + K/B)\ I/Os$ for $k$-dimensional point sets and prove that the resulting structure indeed achieves this query bound.

(b) While a polylogarithmic query bound for orthogonal range reporting cannot be achieved using a linear-space data structure, the query bound achieved by the $k$-d-tree is not the best possible for linear-space orthogonal range reporting data structures. Develop a linear-space orthogonal range reporting data structure that achieves a query bound of $O((N/B)^{\epsilon} + K/B)\ I/Os$ for any constant-dimensional point set. Argue briefly that your structure is correct and achieves the required space and query bounds.

**Note:** It helps to think about two dimensions first and then lift the ideas into higher dimensions. For $k$ dimensions, the constant factors in the query and space bounds will grow proportionally to $(1/\epsilon)^{k-1}$. Thus, this really works only if $k$ is a constant.

(c) Can you make the structure from part (b) support insertions and deletions using amortized $O(\log_B N)\ I/Os$? If so, explain how. If not, argue why not.